### Application software

[Application software](http://en.wikipedia.org/wiki/Application_software) is developed to perform in any task that benefits from computation. It is a set of programs that allows the computer to perform a specific data processing job for the user.It is a broad category, and encompasses **software** of many kinds, including the [internet browser](http://en.wikipedia.org/wiki/Internet_browser) being used to display this page. This category includes:

* [Business software](http://en.wikipedia.org/wiki/Business_software)
* [Computer-aided design](http://en.wikipedia.org/wiki/Computer-aided_design)
* [Databases](http://en.wikipedia.org/wiki/Database)
* [Decision-making software](http://en.wikipedia.org/wiki/Decision-making_software)
* [Educational software](http://en.wikipedia.org/wiki/Educational_software)
* [Image editing](http://en.wikipedia.org/wiki/Image_editing)
* [Industrial automation](http://en.wikipedia.org/wiki/Automation)
* [Mathematical software](http://en.wikipedia.org/wiki/Mathematical_software)
* [Medical software](http://en.wikipedia.org/wiki/Medical_software)
* [Molecular modeling software](http://en.wikipedia.org/wiki/List_of_software_for_molecular_mechanics_modeling)
* [Quantum chemistry and solid state physics software](http://en.wikipedia.org/wiki/List_of_quantum_chemistry_and_solid_state_physics_software)
* [Simulation software](http://en.wikipedia.org/wiki/Simulation_software)
* [Spreadsheets](http://en.wikipedia.org/wiki/Spreadsheet)
* [Telecommunications](http://en.wikipedia.org/wiki/Telecommunication) (i.e., [the Internet](http://en.wikipedia.org/wiki/Internet) and everything that flows on it)
* [Video editing software](http://en.wikipedia.org/wiki/Video_editing_software)
* [Video games](http://en.wikipedia.org/wiki/Video_game)
* [Word processing](http://en.wikipedia.org/wiki/Word_processing)

## Software topics

### Architecture

Users often see things differently than programmers. People who use modern general purpose computers (as opposed to [embedded systems](http://en.wikipedia.org/wiki/Embedded_system), [analog computers](http://en.wikipedia.org/wiki/Analog_computer) and [supercomputers](http://en.wikipedia.org/wiki/Supercomputer)) usually see three layers of software performing a variety of tasks: platform, application, and user software.

* Platform software: [Platform](http://en.wikipedia.org/wiki/Platform_(computing)) includes the [firmware](http://en.wikipedia.org/wiki/Firmware), [device drivers](http://en.wikipedia.org/wiki/Device_driver), an [operating system](http://en.wikipedia.org/wiki/Operating_system), and typically a [graphical user interface](http://en.wikipedia.org/wiki/Graphical_user_interface) which, in total, allow a user to interact with the computer and its [peripherals](http://en.wikipedia.org/wiki/Peripheral) (associated equipment). Platform software often comes bundled with the computer. On a [PC](http://en.wikipedia.org/wiki/Personal_computer) you will usually have the ability to change the platform software.
* Application software: [Application software](http://en.wikipedia.org/wiki/Application_software) or Applications are what most people think of when they think of software. Typical examples include office suites and video games. Application software is often purchased separately from computer hardware. Sometimes applications are bundled with the computer, but that does not change the fact that they run as independent applications. Applications are usually independent programs from the operating system, though they are often tailored for specific platforms. Most users think of compilers, databases, and other "system software" as applications.
* User-written software: [End-user development](http://en.wikipedia.org/wiki/End-user_development) tailors systems to meet users' specific needs. User software include spreadsheet templates and [word processor](http://en.wikipedia.org/wiki/Word_processor) templates. Even email filters are a kind of user software. Users create this software themselves and often overlook how important it is. Depending on how competently the user-written software has been integrated into default application packages, many users may not be aware of the distinction between the original packages, and what has been added by co-workers.

### Documentation

[*Software documentation*](http://en.wikipedia.org/wiki/Software_documentation)

Most software has [software documentation](http://en.wikipedia.org/wiki/Software_documentation) so that the [end user](http://en.wikipedia.org/wiki/End_user) can understand the program, what it does, and how to use it. Without clear documentation, software can be hard to use—especially if it is very specialized and relatively complex like [Photoshop](http://en.wikipedia.org/wiki/Photoshop) or [AutoCAD](http://en.wikipedia.org/wiki/AutoCAD).

Developer documentation may also exist, either with the code as comments and/or as separate files, detailing how the programs works and can be modified.

### Library

[*Software library*](http://en.wikipedia.org/wiki/Software_library)

An executable is almost always not sufficiently complete for direct execution. [Software libraries](http://en.wikipedia.org/wiki/Software_library) include collections of [functions](http://en.wikipedia.org/wiki/Function_(computer_science)) and functionality that may be embedded in other applications. Operating systems include many standard Software libraries, and applications are often distributed with their own libraries.

### Standard

[*Software standard*](http://en.wikipedia.org/wiki/Software_standard)

Since software can be designed using many different [programming languages](http://en.wikipedia.org/wiki/Programming_languages) and in many different [operating systems](http://en.wikipedia.org/wiki/Operating_systems) and [operating environments](http://en.wikipedia.org/wiki/Operating_environment), [software standard](http://en.wikipedia.org/wiki/Software_standard) is needed so that different software can understand and exchange information between each other. For instance, an [email](http://en.wikipedia.org/wiki/Email) sent from a [Microsoft Outlook](http://en.wikipedia.org/wiki/Microsoft_Outlook) should be readable from [Yahoo! Mail](http://en.wikipedia.org/wiki/Yahoo!_Mail) and vice versa.

### Execution

[*Execution (computing)*](http://en.wikipedia.org/wiki/Execution_(computing))

Computer software has to be "loaded" into the [computer's storage](http://en.wikipedia.org/wiki/Computer_storage) (such as the [hard drive](http://en.wikipedia.org/wiki/Hard_drive) or [memory](http://en.wikipedia.org/wiki/Computer_memory)). Once the software has loaded, the computer is able to *execute* the software. This involves passing [instructions](http://en.wikipedia.org/wiki/Instruction_(computer_science)) from the application software, through the system software, to the hardware which ultimately receives the instruction as [machine code](http://en.wikipedia.org/wiki/Machine_language). Each instruction causes the computer to carry out an operation – moving [data](http://en.wikipedia.org/wiki/Data_(computing)), carrying out a [computation](http://en.wikipedia.org/wiki/Computation), or altering the [control flow](http://en.wikipedia.org/wiki/Control_flow) of instructions.

Data movement is typically from one place in memory to another. Sometimes it involves moving data between memory and registers which enable high-speed data access in the CPU. Moving data, especially large amounts of it, can be costly. So, this is sometimes avoided by using "pointers" to data instead. Computations include simple operations such as incrementing the value of a variable data element. More complex computations may involve many operations and data elements together.

### Quality and reliability

[*Software quality*](http://en.wikipedia.org/wiki/Software_quality)*,* [*Software testing*](http://en.wikipedia.org/wiki/Software_testing)*, and* [*Software reliability*](http://en.wikipedia.org/wiki/Software_reliability)

Software quality is very important, especially for commercial and system software like [Microsoft Office](http://en.wikipedia.org/wiki/Microsoft_Office), [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows) and [Linux](http://en.wikipedia.org/wiki/Linux). If software is faulty (buggy), it can delete a person's work, crash the computer and do other unexpected things. Faults and errors are called "[bugs](http://en.wikipedia.org/wiki/Software_bug)." Many bugs are discovered and eliminated (debugged) through [software testing](http://en.wikipedia.org/wiki/Software_testing). However, software testing rarely – if ever – eliminates every bug; some programmers say that "every program has at least one more bug" (Lubarsky's Law). All major software companies, such as Microsoft, Novell and [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems), have their own software testing departments with the specific goal of just testing. Software can be tested through [unit testing](http://en.wikipedia.org/wiki/Unit_testing), [regression testing](http://en.wikipedia.org/wiki/Regression_testing) and other methods, which are done manually, or most commonly, automatically, since the amount of code to be tested can be quite large. For instance, [NASA](http://en.wikipedia.org/wiki/NASA) has extremely rigorous software testing procedures for many operating systems and communication functions. Many NASA based operations interact and identify each other through command programs called software. This enables many people who work at NASA to check and evaluate functional systems overall. Programs containing command software enable hardware engineering and system operations to function much easier together.

### License

[*Software license*](http://en.wikipedia.org/wiki/Software_license)

The software's license gives the user the right to use the software in the licensed environment. Some software comes with the license when purchased off the shelf, or an OEM license when bundled with hardware. Other software comes with a [free software license](http://en.wikipedia.org/wiki/Free_software_license), granting the recipient the rights to modify and redistribute the software. Software can also be in the form of [freeware](http://en.wikipedia.org/wiki/Freeware) or [shareware](http://en.wikipedia.org/wiki/Shareware).

### Patents

[*Software patent*](http://en.wikipedia.org/wiki/Software_patent) *and* [*Software patent debate*](http://en.wikipedia.org/wiki/Software_patent_debate)

Software can be patented in some but not all countries; however, [software patents](http://en.wikipedia.org/wiki/Software_patent) can be controversial in the software industry with many people holding different views about it. The controversy over software patents is about specific [algorithms](http://en.wikipedia.org/wiki/Algorithms) or techniques that the software contains, which may not be duplicated by others and considered [intellectual property](http://en.wikipedia.org/wiki/Intellectual_property) and [copyright infringement](http://en.wikipedia.org/wiki/Copyright_infringement) depending on the severity.

## Design and implementation

[*Software development*](http://en.wikipedia.org/wiki/Software_development)*,* [*Computer programming*](http://en.wikipedia.org/wiki/Computer_programming)*, and* [*Software engineering*](http://en.wikipedia.org/wiki/Software_engineering)

Design and implementation of software varies depending on the complexity of the software. For instance, design and creation of [Microsoft Word](http://en.wikipedia.org/wiki/Microsoft_Word) software will take much more time than designing and developing [Microsoft Notepad](http://en.wikipedia.org/wiki/Microsoft_Notepad) because of the difference in functionalities in each one.

Software is usually designed and created (coded/written/programmed) in [integrated development environments](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) like [Eclipse](http://en.wikipedia.org/wiki/Eclipse_(software)), [Emacs](http://en.wikipedia.org/wiki/Emacs) and [Microsoft Visual Studio](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio) that can simplify the process and [compile](http://en.wikipedia.org/wiki/Compiler) the program. As noted in different section, software is usually created on top of existing software and the [application programming interface](http://en.wikipedia.org/wiki/Application_programming_interface) (API) that the underlying software provides like [GTK+](http://en.wikipedia.org/wiki/GTK%2B), JavaBeans or [Swing](http://en.wikipedia.org/wiki/Swing_(Java)). Libraries (APIs) are categorized for different purposes. For instance, [JavaBeans](http://en.wikipedia.org/wiki/JavaBeans) library is used for designing [enterprise applications](http://en.wikipedia.org/wiki/Enterprise_application), [Windows Forms](http://en.wikipedia.org/wiki/Windows_Forms) library is used for designing graphical user interface (GUI) applications like [Microsoft Word](http://en.wikipedia.org/wiki/Microsoft_Word), and [Windows Communication Foundation](http://en.wikipedia.org/wiki/Windows_Communication_Foundation) is used for designing [web services](http://en.wikipedia.org/wiki/Web_services). Underlying [computer programming](http://en.wikipedia.org/wiki/Computer_programming) [concepts](http://en.wikipedia.org/wiki/Concept) like [quicksort](http://en.wikipedia.org/wiki/Quicksort), [hashtable](http://en.wikipedia.org/wiki/Hashtable), [array](http://en.wikipedia.org/wiki/Array_data_type), and [binary tree](http://en.wikipedia.org/wiki/Binary_tree) can be useful to creating software. When a program is designed, it relies on the API. For instance, if a user is designing a Microsoft Windows desktop application, he/she might use the [.NET](http://en.wikipedia.org/wiki/.NET_Framework) Windows Forms library to design the desktop application and call its APIs like *Form1.Close()* and *Form1.Show()*[[6]](http://en.wikipedia.org/wiki/Software#cite_note-5) to close or open the application and write the additional operations him/herself that it need to have. Without these APIs, the programmer needs to write these APIs him/herself. Companies like [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems), [Novell](http://en.wikipedia.org/wiki/Novell), and [Microsoft](http://en.wikipedia.org/wiki/Microsoft) provide their own APIs so that many applications are written using their [software libraries](http://en.wikipedia.org/wiki/Software_library) that usually have numerous APIs in them.

Computer software has special economic characteristics that make its design, creation, and distribution different from most other economic goods.[[7]](http://en.wikipedia.org/wiki/Software#cite_note-6)[[8]](http://en.wikipedia.org/wiki/Software#cite_note-7) A person who creates software is called a [programmer](http://en.wikipedia.org/wiki/Programmer), [software engineer](http://en.wikipedia.org/wiki/Software_engineer), [software developer](http://en.wikipedia.org/wiki/Software_developer), or [code monkey](http://en.wikipedia.org/wiki/Code_monkey), terms that all have a similar meaning.

Software types

Examples of computer software include:

* [Application software](http://en.wikipedia.org/wiki/Application_software) includes end-user applications of computers such as [word processors](http://en.wikipedia.org/wiki/Word_processors) or [video games](http://en.wikipedia.org/wiki/Video_game), and [ERP software](http://en.wikipedia.org/wiki/ERP_software) for groups of users.
* [Middleware](http://en.wikipedia.org/wiki/Middleware_(distributed_applications)) controls and co-ordinates [distributed systems](http://en.wikipedia.org/wiki/Distributed_computing).
* [Programming languages](http://en.wikipedia.org/wiki/Programming_language) define the [syntax](http://en.wikipedia.org/wiki/Syntax) and [semantics](http://en.wikipedia.org/wiki/Semantics) of computer programs. For example, many mature banking applications were written in the [COBOL](http://en.wikipedia.org/wiki/COBOL) language, originally invented in 1959. Newer applications are often written in more modern programming languages.
* [System software](http://en.wikipedia.org/wiki/System_software) includes [operating systems](http://en.wikipedia.org/wiki/Operating_system), which govern computing resources. Todaylargeapplications running on remote machines such as [Websites](http://en.wikipedia.org/wiki/Website) are considered to be system software, becausethe end-[user interface](http://en.wikipedia.org/wiki/User_interface) is generally through a [graphical user interface](http://en.wikipedia.org/wiki/Graphical_user_interface), such as a [web browser](http://en.wikipedia.org/wiki/Web_browser).
* *Teachware* is any special breed of software or other means of product dedicated to education purposes in software engineering and beyond in general education.
* [Testware](http://en.wikipedia.org/wiki/Testware) is any software for testing hardware or software.
* [Firmware](http://en.wikipedia.org/wiki/Firmware) is low-level software often stored on electrically programmable memory devices. Firmware is given its name because it is treated like hardware and run ("executed") by other software programs. Firmware often is not accessible for change by other entities but the developers' enterprises.
* Shrinkware is the older name given to consumer-purchased software, because it was often sold in retail stores in a [shrink-wrapped](http://en.wikipedia.org/wiki/Shrink_wrap) box.
* [Device drivers](http://en.wikipedia.org/wiki/Device_drivers) control parts of computers such as [disk drives](http://en.wikipedia.org/wiki/Disk_drive), [printers](http://en.wikipedia.org/wiki/Printer_(computing)), [CD drives](http://en.wikipedia.org/wiki/CD_drive), or [computer monitors](http://en.wikipedia.org/wiki/Computer_monitor).
* [Programming tools](http://en.wikipedia.org/wiki/Programming_tool) help conduct computing tasks in any category listed above. For programmers, these could be tools for [debugging](http://en.wikipedia.org/wiki/Debugging) or [reverse engineering](http://en.wikipedia.org/wiki/Reverse_engineering) older [legacy systems](http://en.wikipedia.org/wiki/Legacy_system) in order to check [source code compatibility](http://en.wikipedia.org/wiki/Source_code_compatibility).

**Industry and organizations**

*Main article:* [*Software industry*](http://en.wikipedia.org/wiki/Software_industry)

A great variety of software companies and programmers in the world comprise a software industry. Software can be quite a profitable industry: [Bill Gates](http://en.wikipedia.org/wiki/Bill_Gates), the founder of [Microsoft](http://en.wikipedia.org/wiki/Microsoft) was the richest person in the world in 2009 largely by selling the [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows) and [Microsoft Office](http://en.wikipedia.org/wiki/Microsoft_Office) software products. The same goes for [Larry Ellison](http://en.wikipedia.org/wiki/Larry_Ellison), largely through his [Oracle database](http://en.wikipedia.org/wiki/Oracle_database) software. Through time the software industry has become increasingly specialized.

Non-profit software organizations include the [Free Software Foundation](http://en.wikipedia.org/wiki/Free_Software_Foundation), [GNU Project](http://en.wikipedia.org/wiki/GNU_Project) and [Mozilla Foundation](http://en.wikipedia.org/wiki/Mozilla_Foundation). Software standard organizations like the [W3C](http://en.wikipedia.org/wiki/W3C), [IETF](http://en.wikipedia.org/wiki/IETF) develop software standards so that most software can interoperate through standards such as [XML](http://en.wikipedia.org/wiki/XML), [HTML](http://en.wikipedia.org/wiki/HTML), [HTTP](http://en.wikipedia.org/wiki/HTTP) or [FTP](http://en.wikipedia.org/wiki/FTP).

Other well-known large software companies include [Novell](http://en.wikipedia.org/wiki/Novell), [SAP](http://en.wikipedia.org/wiki/SAP_AG), [Symantec](http://en.wikipedia.org/wiki/Symantec), [Adobe Systems](http://en.wikipedia.org/wiki/Adobe_Systems), and [Corel](http://en.wikipedia.org/wiki/Corel), while small companies often provide innovation.